Elsevier LaTeX template[⋆]

Elsevier[1]

Radarweg 29, Amsterdam

Elsevier Inc[a,b], Global Customer Service[b,]*

[a]1600 John F Kennedy Boulevard, Philadelphia
[b]360 Park Avenue South, New York

Abstract

This template helps you to create a properly formatted LaTeX manuscript.

Keywords: `elsarticle.cls`, LaTeX, Elsevier, template

2010 MSC: 00-01, 99-00

1. Introduction

Since the dawn of internet and world wide web, humanity has witnessed a degree of connection beyond reckoning. The proliferation of digital devices pervaded with various applications that account for almost all aspect of humanity, have created cyber communities that constantly mutate [1]; [2]. In a world where we have network infrastructures that can support up to 250Mbps of data transmission, and smart phones and IOT devices that can have processing power of up to 3 Ghz, data becomes ubiquitous, the quantum that lays the foundation of the nexus [3].

According to InternetLiveStates.com [4], only in one second, there are 9,878 tweets sent, 1,138 instagram photos uploaded, 3,117,720 emails sent, 99,738 Google searches made, and 94,144 Youtube videos viewed. That is, if it has

[⋆]Fully documented templates are available in the elsarticle package on CTAN.

[*]Corresponding author
Email address: `support@elsevier.com` (Global Customer Service)
URL: `www.elsevier.com` (Elsevier Inc)

[1]Since 1880.

taken 5 second the read the preceding paragraph, during that time, 15,588,600 emails are sent.

Driven by the ambition to harness the power of this deluge of data, the term 'Big Data' (BD) was coined [5]. BD initially emerged to address the challenges associated with various characteristics of data such as velocity, variety, volume and variability [2]. BD is the practice of extracting patterns, theories, and predictions from a large set of structured, semi-structured, and unstructured data for the purposes of business competitive advantage [6]; [7]. BD is a game-changing innovation, heralding the dawn of a new data-oriented industry.

Nonetheless, BD is not a magical wand that can enchant any business process. While a lot of opportunities exist in BD, subsuming an emergent and rather high-impacting technology like BD to current state of affairs in organizations, is a daunting task. According to recent survey from Databricks, only 13% of the organizations excel at delivering on their data strategy [8]. Another survey by NewVantage Partners indicated that only 24% organization have successfully gone data-driven [9]. This survey also states that only 30% of organizations have a well established strategy for their big data endeavour. In addition, surveys from McKinsey & Company ([10]) and Gartner ([11]) further support these numbers, which illuminates on the scarcity of successful big data implementations in the industry.

Among the challenges of data adoption perhaps the most highlighted are 'data engineering complexities', 'big data architecture', 'rapid technology change', 'lack of sufficient skilled data engineers', and 'organization's cultural challenges of becoming data-driven' [2];[12]. This focus of this study is on data engineering complexities and in specific big data architecture.

In the past, organization relied on a few technology giants to provide infrastructure and tools necessary for big data, while today there's a plethora of choice from hundreds of providers covering different aspect of data ecosystem from ingestion, to logging, to stream processing, and to visualization [9]. Companies are tending more and more towards Cloud-native architectures for cost reduction, improved efficiency and new roles have been introduced such as chief analytics officer (CAOs) amd chief data officers (CDOs) to channel the organizational big data capabilities toward business value and competitive advantage [13].

So how can one embark on this rather sophisticated journey? what can be a good logical approach to absorb the ever-increasing complexity of big data systems? how can organizations build different stacks to handle data for various workloads such as machine learning (ML), business analytics, data engineering, and streaming?

We suggest that majority of the challenge discussed starts with data architecture [1]; [3]. The data ingestion, processing and consumption of different data workloads vary, and sometimes they don't go well together. A company that enacted a data lake and a data warehouse and tries to account for both ecosystems, can be dealing with immense complexity, which in turns impact data teams, which in turn can hinder innovation, create barriers and result in monumental lost.

Development and deployment of an efficacious big data system is only the beginning of a big data journey. As data sources increase, variety of data increases, number of data consumers increase, the data store gets confuscated, and this can introduce threats for scalability and maintainability of the system. This also implies that only a handful of hyper-specialized data engineers would understand the system internals, creating silos, and potential miscommunication.

Majority of these systems are developed on-premise as ad-hoc complicated solutions that do not adhere to the practices of software engineering and software architecture [14]; [15]. As the ecosystem grows and new technologies and data processing techniques are introduced, the software architect will have a harder time to come up with a solution that address the problem requirements.

This can potentially create grounds for an immature architecture that results in solutions that are hard to scale, hard to maintain, and raise high-entry blockades [3]. Since the approach of ad-hoc design to big data system development is not desirable and may leave many architects and data engineers in the dark, novel data architectures that are designed specifically for BD are required. To contribute to this goal, we explore the notion of reference architectures (RAs) and present a distributed domain-driven software RA for big data systems.

2. Why reference architecture?

To justify why we have chosen reference architectures as the suitable artefact, first we have to clarify two assumptions;

1. having a sound software architecture is essential to the successful development and maintenance of software systems
2. there exist a sufficient body of knowledge in the field of software architecture to support the development of an effective RA

One of the focal tenets of software architecture is that every system is developed to satisfy a business objective, and that the architecture of the system is a bridge between abstract business goals to concrete final solutions [16]. While the journey of big data can be quite challenging, the good news is that a software RA can be designed, analyzed and documented incorporating best practices, known techniques, and patterns that will support the achievement of the business goals. In this way, the complexity can be absorbed, and made tractable.

Practitioners of complex systems, software engineers, and system designers have been frequently using reference architectures to have a collective understanding of system components, functionalities, data-flows and patterns which shape the overall qualities of system and help further adjust it to the business objectives [17]; [18]. There is a fair amount of literature on reference architectures, and whereas different authors definition may vary, they all share the same tenets.

A reference architecture is amalgamation of architectural patterns, standards, software engineering techniques that bridge the problem domain to a class of solutions. This artefact can be partially or completely instantiated and prototyped in a particular business context together with other supporting artefact to enable its use. RAs are often created from previous RAs and architecture [1].

The usage of RAs for the development of complex systems is not new. In software product line (SPL) development, RAs are generic artifacts that are configured and instantiated for a particular domain of systems [19]. In software engineering, major IT giants like IBM has referred to RAs as the 'best of best practices' to address unique and complex system development challenges [17].

Based on the premises discussed and taking all into consideration, RAs can facilitate the issues of big data architecture and data engineering because of the following reasons;

1. RAs can promote adherence to best practice, standards, specifications and patterns
2. RAs can endow the data architecture team with openness and increase operability, incorporating architectural patterns that ensue desirable predefined quality attributes
3. RAs can be the best initial start to the big data journey, capturing design issues when they are still cheap
4. RAs can bring different stakeholders on the same table and help achieve consensus around major technological constructs
5. RAs can be effective in identifying and addressing cross-cutting concerns
6. RAs can serve as the organizational memory around design decisions, enlightening next subsequent decisions
7. RAs can act as a summary and blueprint in the portfolio of software engineers and architect, resulting in better dissemination of knowledge

3. Research Methodology

There are a few studies that have addressed the systematic development of reference architectures. Cloutier et al [17] present a high-level model for RA development through collection of contemporary information and capturing the essence of architectural advancements. In another effort,PuLSE-DSSA is proposed by Bayer et al. [20] in the context of product line development and domain engineering. PulSE-DSSA emphasizes on capturing the existing architectural knowledge. Stricker et al. [21] propose a pattern-based approach for creating an RA. This study revolves around software engineering patterns motivated by the work of Gamma et al [22]; proposing a structural approach that includes three layers of patterns with well-defined hierarchical relationships. Nakagawa, Martins, Felizardo, and Maldonado [23] propose an approach to RA design outside of product line management context that is concentrated towards aspect-oriented systems.

Galster and Avgeriou [24] propose an empirically grounded reference architecture based on two main facets; Existing RAs in practice and available literature on RAs. Along the same vein, Nakagawa et al [25] presented ProSA-RA which is a 4 phase methodology that unlike many other methodologies do provide a more comprehensive instructions on RA evaluation. In addition, this methodology benefits from an ecosystem of complementary constructs that aid in RA design and evaluation such as RAModel [26] and a framework for evaluation of RAs (FERA) [27].In a recent study, Derras et al. [28] propose a schema of practical RA development in the context of software product line and domain engineering. This study is based on capturing knowledge from architectures in practice with attention to variability, configurability and product line development. The findings provide a four-phase process to develop quality driven reference architectures. This approach is influenced by ISO/IEC 26550 [29].

By analysis and study of all these approaches for design and development of RAs, a common pattern has been witnessed. Whereas some of them are more recent and some belong to years ago, there are commonalities that has been observed. All these approaches are grounded on three main pillars, 1) Existing RAs 2) RAs in literature 3) Architectures in practice. Taking this into consideration and by analyzing the results of the systematic literature review conducted by Ataei et al [1] we found'Empirically-grounded reference architectures' proposed by Galster and Avgeriou [24], a suitable methodology, because firstly it's been adopted by many studies, and secondly it's comparatively in-line with the nature of our study.

Nevertheless, we did not fully adopt this methodology and rather customized to the needs of this particular research. This is due to some inherent limitations that has been witnessed with the methodology. For instance we could not find a comprehensive guideline on how to identify data sources and how it could be categorized and synthesized into the creation of the RA in the third step of the methodology, therefore we employed the Nakagawa's information source investigation guidelines and the overall idea of the RAModel. Another limitation we've faced was with evaluation of the RA. As evaluation, second to a sound research methodology is one of the key elements of any good design science research, we had to look for a stronger and more systematic evaluation approach than what was discussed in 'empirically grounded RAs' methodology. For this purpose, and inspired by the works of Angelov et al [30]; [31], we first created an prototype of the RA in practice, and then used 'The architecture tradeoff analysis method' (ATAM) [**?**] to evaluate the artefact.

This research methodology is constituent of 6 phases which are respectively; 1) Decision on the type of the RA 2) Design strategy 3) Empirical acquisition of data 4) Construction of the RA 5) Enable RA with variability 6) Evaluation of the RA. The phrase 'empirically grounded' refers to two major elements; firstly the reference architecture should be grounded in well-established and proven principles; secondly, the reference architecture should be evaluated for applicability and validity. These don't only belong to Galster and Avgeriou methodology, and other researchers such as Cloutier [17] and Derras et al [19] have promoted the same ideas.

It is worth mentioning that this methodology is iterative, meaning that the results gained from the evaluation phase (6th phase) determines the subsequent iterations until the design reaches saturation.

3.1. Decision on type of the RA

Precursor to any effective RA development, is the decision on type of it. The type of the RA is significant, as it illuminates on information to be collected and the construction of the RA in later phases. The selection on the type of RA for the purposes of this study is based on two dimensions; the classification framework proposed by Angelov et al. [32] and the usage context [33].

Based on the classification framework proposed by Angelov et al. [32], five types of RA are defined. This framework has been developed with the goal of supporting analysis of RAs with regards to context, goal, and the architecture specification/design relationships. It is based on 3 major dimensions namely context, goals, and design, each having their own corresponding sub-dimensions. These dimensions and sub-dimensions are derived by the means of interrogatives (the usage of interrogates is a well-established practice for problem analysis (the usage of interrogates is a well-established practice for problem analysis).

The interrogatives 'When', 'Where', and 'Who' have been used to address the 'context', 'Why' has been used to address 'goal', and 'How' and 'What' have been used to address 'design' dimension. The outcome of the study categorizes RAs in two major groups; 1) standardization RAs and 2) Facilitation RAs. This framework has been chosen because it is completely in-line with the purposes of this study and aims to demarcate a clear domain for the RA to be developed. The comprehensive classification of the RAs with examples in practice illuminates on how different RAs are playing roles in the industry and how they are classified. This brings clarity on what should be developed and what boundaries should be drawn.

By reading the results of the recent SLR conducted by Ataei et al on BD RAs [1], we've added more examples of the RAs on top of what was provided by Angelov [32], and provided the following updated list of RA classifications with examples;

1. Standardization RAs
    (a) Type 1: classical, standardization architectures designed to be implemented in multiple organizations. Examples are:
        i. WRM [34]
        ii. OSI RM [35]

iii. OATH [36]
iv. COBRA [37]
v. Neomycelia [3]
vi. Kappa [38]
vii. Bolster [15]

(b) Type 2: classical, standardization architectures designed to be implemented in a single organization

i. Fortis Bank Reference Software Architecture [?]

2. Facilitation RAs

(a) Type 3: classical, facilitation reference architectures for multiple organizations designed by a software organization in cooperation with user organizations

i. Microsoft Application Architecture for .Net [39]
ii. IBM PanDOORA
iii. OATH [36]
iv. COBRA [37]

(b) Type 4: classical, facilitation architectures designed to be implemented in a single organization

i. Achmea Software Reference Architecture [40]
ii. ABN-AMRO Web Application Architecture [41]

(c) Type 5: preliminary, facilitation architectures designed to be implemented in multiple organizations

i. ERA [33]
ii. AHA [42]
iii. eSRA [43]

The domain driven distributed BD RA chosen for the purposes of this study pursues two major goals; 1) enabling and support the development and data engineering of big data systems 2) concurrently ensuring that interoperability between different heterogeneous components of the big data system is established. Therefore, the outcome artefact will be a BD RA that is a classical standardization RA designed to be implemented in multiple organizations.

References

References

[1] P. Ataei, A. T. Litchfield, Big data reference architectures, a systematic literature review.

[2] B. B. Rad, P. Ataei, The big data ecosystem and its environs, International Journal of Computer Science and Network Security (IJCSNS) 17 (3) (2017) 38.

[3] P. Ataei, A. Litchfield, Neomycelia: A software reference architecturefor big data systems, in: 2021 28th Asia-Pacific Software Engineering Conference (APSEC), IEEE Computer Society, Los Alamitos, CA, USA, 2021, pp. 452–462. `doi:10.1109/APSEC53868.2021.00052`.
URL `https://doi.ieeecomputersociety.org/10.1109/APSEC53868.2021.00052`

[4] I. L. Stats, Internet live stats (2019).

[5] M. Lycett, 'datafication': Making sense of (big) data in a complex world (2013).

[6] B. B. Rada, P. Ataeib, Y. Khakbizc, N. Akbarzadehd, The hype of emerging technologies: Big data as a service.

[7] M. Huberty, Awaiting the second big data revolution: from digital noise to value creation, Journal of Industry, Competition and Trade 15 (1) (2015) 35–47.

[8] M. technology review insights in partnership with Databricks, Building a high-performance data organization (2021).
URL `https://databricks.com/p/whitepaper/mit-technology-review-insights-report`

[9] N. Partners, Big data and ai executive survey 2021 (2021). URL https://www.supplychain247.com/paper/big_data_and_ai_executive_survey_2021/pragmadik

[10] M. Analytics, The age of analytics: competing in a data-driven world, Tech. rep., Technical report, San Francisco: McKinsey & Company (2016).

[11] H. Nash, Cio survey 2015, Association with KPMG.

[12] N. Singh, K.-H. Lai, M. Vejvar, T. Cheng, Big data technology: Challenges, prospects and realities, IEEE Engineering Management Review.

[13] B. B. Rad, P. Ataei, Evaluating major issues regarding reliability management for cloud-based applications, IJCSNS 17 (7) (2017) 168.

[14] I. Gorton, J. Klein, Distribution, data, deployment, STC 2015 (2015) 78.

[15] S. Nadal, V. Herrero, O. Romero, A. Abelló, X. Franch, S. Vansummeren, D. Valerio, A software reference architecture for semantic-aware big data systems, Information and software technology 90 (2017) 75–92.

[16] R. K. Len Bass, Dr. Paul Clements, Software Architecture in Practice (SEI Series in Software Engineering) 4th Edition, Addison-Wesley Professional; 4th edition, 2021.

[17] R. Cloutier, G. Muller, D. Verma, R. Nilchiani, E. Hole, M. Bone, The concept of reference architectures, Systems Engineering 13 (1) (2010) 14–27.

[18] J. Kohler, T. Specht, Towards a secure, distributed, and reliable cloud-based reference architecture for big data in smart cities, in: Big Data Analytics for Smart and Connected Cities, IGI Global, 2019, pp. 38–70.

[19] M. Derras, L. Deruelle, J.-M. Douin, N. Levy, F. Losavio, Y. Pollet, V. Reiner, Reference architecture design: A practical approach, in: ICSOFT, pp. 633–640.

[20] J. Bayer, T. Forster, D. Ganesan, J.-F. Girard, I. John, J. Knodel, R. Kolb, D. Muthig, Definition of reference architectures based on existing systems, Fraunhofer IESE, March.

[21] V. Stricker, K. Lauenroth, P. Corte, F. Gittler, S. De Panfilis, K. Pohl, Creating a reference architecture for service-based systems–a pattern-based approach, in: Towards the Future Internet, IOS Press, 2010, pp. 149–160.

[22] E. Gamma, R. Helm, R. Johnson, R. E. Johnson, J. Vlissides, et al., Design patterns: elements of reusable object-oriented software, Pearson Deutschland GmbH, 1995.

[23] E. Y. Nakagawa, R. M. Martins, K. R. Felizardo, J. C. Maldonado, Towards a process to design aspect-oriented reference architectures, in: XXXV Latin American Informatics Conference (CLEI) 2009, 2009.

[24] M. Galster, P. Avgeriou, Empirically-grounded reference architectures: a proposal, in: Proceedings of the joint ACM SIGSOFT conference–QoSA and ACM SIGSOFT symposium–ISARCS on Quality of software architectures–QoSA and architecting critical systems–ISARCS, 2011, pp. 153–158.

[25] E. Y. Nakagawa, M. Guessi, J. C. Maldonado, D. Feitosa, F. Oquendo, Consolidating a process for the design, representation, and evaluation of reference architectures, in: 2014 IEEE/IFIP Conference on Software Architecture, IEEE, 2014, pp. 143–152.

[26] E. Y. Nakagawa, F. Oquendo, M. Becker, Ramodel: A reference model for reference architectures, in: 2012 Joint Working IEEE/IFIP Conference on Software Architecture and European Conference on Software Architecture, IEEE, 2012, pp. 297–301.

[27] J. F. M. Santos, M. Guessi, M. Galster, D. Feitosa, E. Y. Nakagawa, A checklist for evaluation of reference architectures of embedded systems (s)., in: SEKE, Vol. 13, 2013, pp. 1–4.

[28] M. Derras, L. Deruelle, J. M. Douin, N. Levy, F. Losavio, Y. Pollet, V. Reiner, Reference architecture design: a practical approach, in: 13th International Conference on Software Technologies (ICSOFT), SciTePress-Science and Technology Publications, 2018, pp. 633–640.

[29] I. WG, Iso/iec 26550: 2015-software and systems engineering–reference model for product line engineering and management, ISO/IEC, Tech. Rep.

[30] S. Angelov, J. J. Trienekens, P. Grefen, Towards a method for the evaluation of reference architectures: Experiences from a case, in: European Conference on Software Architecture, Springer, 2008, pp. 225–240.

[31] S. Angelov, J. J. Trienekens, P. Grefen, Extending and adapting the architecture tradeoff analysis method for the evaluation of software reference architectures.

[32] R. Kazman, M. Klein, M. Barbacci, T. Longstaff, H. Lipson, J. Carriere, The architecture tradeoff analysis method, in: Proceedings. fourth ieee international conference on engineering of complex computer systems (cat. no. 98ex193), IEEE, 1998, pp. 68–78.

[33] S. Angelov, P. Grefen, D. Greefhorst, A classification of software reference architectures: Analyzing their success and effectiveness, in: 2009 Joint Working IEEE/IFIP Conference on Software Architecture & European Conference on Software Architecture, IEEE, 2009, pp. 141–150.

[34] S. Angelov, P. Grefen, An e-contracting reference architecture, Journal of Systems and Software 81 (11) (2008) 1816–1844.

[35] D. Hollingsworth, U. Hampshire, Workflow management coalition: The workflow reference model, Document Number TC00-1003 19 (16) (1995) 224.

[36] H. Zimmermann, Osi reference model-the iso model of architecture for open systems interconnection, IEEE Transactions on communications 28 (4) (1980) 425–432.

[37] OATH, Oath reference architecture, release 2.0 initiative for open authentication, OATH.
URL https://openauthentication.org/wp-content/uploads/2015/09/ReferenceArchitectureVersion2.pdf

[38] A. L. Pope, The CORBA reference guide: understanding the common object request broker architecture, Addison-Wesley Longman Publishing Co., Inc., 1998.

[39] J. Kreps, Questioning the lambda architecture, Online article, July 205.

[40] M. Press, L. Joyner, G. Malcolm, Application Architecture for. NET: Designing Applications and Services, Microsoft Press, 2002.

[41] D. Greefhorst, P. Gehner, Achmea streamlines application development and integration, Via Nova Architectura.

[42] D. Greefhorst, Een applicatie-architectuur voor het web bij de bank—de pro's en contra's van toestandsloosheid, Software Release Magazine 2.

[43] H. Wu, A reference architecture for Adaptive Hypermedia Applications, Citeseer, 2002.

[44] A. H. Norta, Exploring dynamic inter-organizational business process collaboration (2007).